

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-235116

(43)Date of publication of application : 13.09.1996

(51)Int.Cl.

G06F 15/00

G06F 15/16

(21)Application number : 07-322822

(71)Applicant : INTERNATL BUSINESS MACH CORP <IBM>

(22)Date of filing : 12.12.1995

(72)Inventor : FORTINSKY MICHAEL S

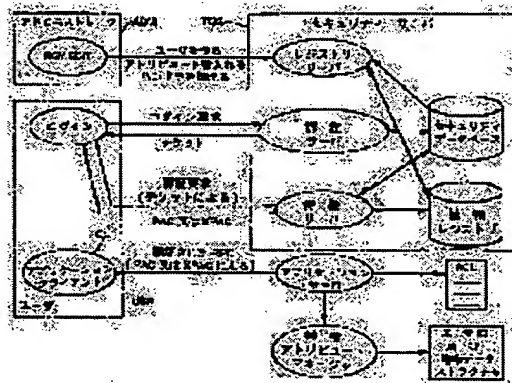
(30)Priority

Priority number : 94 2138302 Priority date : 15.12.1994 Priority country : CA

(54) MECHANISM FOR PROVIDING SAFETY PROTECTIVE ACCESS TO EXTERNAL RESOURCE FROM DISTRIBUTED COMPUTING ENVIRONMENT**(57)Abstract:**

PROBLEM TO BE SOLVED: To provide a safe protective access to an external resource from a distributed computing environment.

SOLUTION: In a distributed computing environment in which a ticket is issued to a client from a security server TGS when the client performs access to a server, the client can access a resource on the outside of the environment upon receiving an extension certificate containing additional data from the security server TGS. The additional data give the information on the privileged attribute, etc., of the client in a format acceptable by an external server, recognized by a server which performs access to the external server, and are transmitted in the environment in a format equivalent to that of the normal ticket. The security server TGS has a registry which is extended to contain the data on the privileged attribute of the client together with the data on the structure in which the data on the privileged attribute must be given with respect to accessible external servers.

**LEGAL STATUS**

[Date of request for examination] 07.11.1997

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration] withdrawal

[Date of final disposal for application] 25.03.2002

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(11)特許出願公開番号

(43)公開日 平成8年(1996)9月13日

(51)Int.Cl. ⁶	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 15/00	3 3 0	9364-5L	G 0 6 F 15/00	3 3 0 C
15/16	3 7 0		15/16	3 7 0 N

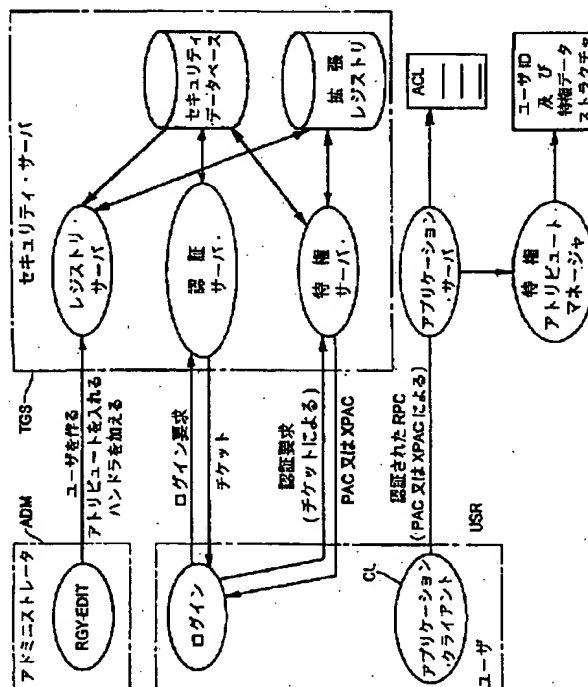
審査請求 未請求 請求項の数4 OL (全 19 頁)

(33)優先権主張国 カナダ (CA)

(74)代理人 弁理士 合田 潔 (外2名)

(54) 【発明の名称】 分散型計算環境から外部資源への安全保護アクセスを提供する機構

【解決手段】サーバをアクセスするためにクライアントがセキュリティ・サーバによってチケットを発行される分散型計算環境において、その環境の外部の資源へのアクセスをセキュリティ・サーバが追加データを含む拡張証明を発行することによって行う。その追加データは、クライアントの特権アトリビュート等に関する情報を、外部サーバにとって受容可能なフォーマットで与えるようにし、又、その追加データは、外部サーバへのアクセスを行うサーバによって認識され、正規のチケットと同等のフォーマットでその環境内で伝送される。セキュリティ・サーバは、アクセス可能な外部サーバに関して、クライアントの特権アトリビュートに関するデータを、そのようなデータが与えられるべきストラクチャに関するデータと共に含むよう拡張されたレジストリを有する。



1

【 特許請求の範囲】

【 請求項1 】 クライアント・アイデンティティ及びアトリビュートと環境における資源に関連したクライアント特権とに関するデータを含んだアトリビュート・レジストリを有するセキュリティ・サーバと、環境の外部の資源へのアクセスを行い且つ環境のセキュリティ要件とは互換性のないセキュリティ要件を有する該環境内の少なくとも1つのアプリケーション・サーバを含むタイプの分散型計算環境にして、前記セキュリティ・サーバは前記環境内のサーバによるサービスを必要とするクライアントに対して要求に応じてチケットを発行し、前記チケットは、前記環境内のクライアントのアイデンティティ及び特権アトリビュートに関する情報を与えるように、サーバへの供給時にデコード可能であるコード化データを含む特権アトリビュート証明を有する分散型計算環境において、

前記セキュリティ・サーバは、前記外部の資源のうちの少なくとも1つに関するクライアント・アイデンティティ及び特権アトリビュートに関する追加情報と各外部の資源が前記追加情報を必要とするというストラクチャに関するデータとを含む拡張レジストリと、サーバが外部の資源へのアクセスを行うことによるサービスのためにクライアントによってリクエストされたチケットに前記追加情報を更なるコード化データとして含むための手段とを有すること、及び外部の資源へのアクセスを行うサーバは、更なるコード化データを認識するための手段と、該認識されたデータをデコードし、外部資源へのアクセスのために必要なストラクチャに該デコードされたデータを配置するための手段とを有することを特徴とする分散型計算環境。

【 請求項2 】 前記セキュリティ・サーバ及び前記外部の資源へのアクセスを行うサーバは前記セキュリティ・サーバの前記証明における前記追加情報を含むアトリビュート・ハンドラを含み、外部の資源へのアクセスのための構造化データを必要とするサーバにおいて前記追加情報をデコード及び構造化することを特徴とする請求項1に記載の分散型計算環境。

【 請求項3 】 前記更なるコード化データは前記環境内のクライアントの特権アトリビュートに関する結果のコード化データに続く単一のデータ・エレメントに含まれることを特徴とする請求項1に記載の分散型計算環境。

【 請求項4 】 サーバへのアクセスを望んでいるクライアントにチケットを発行するためのセキュリティ・サーバと、環境の外部の少なくとも1つの資源にアクセスすることができる少なくとも1つのアプリケーション・サーバを含むタイプの分散型計算環境に対する拡張にして、前記チケットはクライアントのアイデンティティ及び特権アトリビュートに関するコード化情報を含む特権アトリビュート証明を含む分散型計算環境に対する拡張機構において、

2

前記セキュリティ・サーバにおけるデータベースからの、クライアントのアイデンティティ及び特権アトリビュートに関する追加のコード化データ及びそのようなデータが前記環境の外部のサーバに与えられるストラクチャに関する追加のコード化データをストラクチャ内に含むべく特権アトリビュート証明が拡張されるチケットを発行するように前記セキュリティ・サーバを再構成するための手段と、

前記拡張された特権アトリビュート証明を認識するように、前記証明から前記追加データをデコードするように、及び前記外部のサーバに表示するためのデータを構造化するように、前記アプリケーション・サーバのセキュリティ・モジュールを再構成するための手段と、を含む分散型計算環境に対する拡張機構。

【 発明の詳細な説明】

【 0001 】

【 発明の属する技術分野】 本発明は、分散計算を行うコンピュータ・ネットワークにおけるセキュリティに関するものであり、更に詳しく云えば、そのようなネットワークにおいてクライアントによるアクセスを可能にすること、及びそのネットワークからアクセス可能な資源に対するチケット・ベースのセキュリティ・システムを利用するが、異なるベースのセキュリティ・システム或いは互換性のないセキュリティ・システムを利用することに関するものである。

【 0002 】

【 従来の技術】 分散型計算システムは、分散したアプリケーション及びデータの共用を行うためにネットワークを介してコミュニケーションする2つ又はそれ以上の計算機の編成を含んでいる。分散型計算システムの一例は、異種の計算機及びオペレーティング・システムに関連する分散計算をサポートするために、オープン・ソフトウェア・ファウンデーション(Open Software Foundation、以下、OSFという)によってリリースされた分散計算環境である。OSFの分散計算環境(以下、DCEという)は、マサチューセッツ工科大学(MIT)においてその機構のアテナ(Athena) プロジェクトの一部として開発されたカーベロス(Kerberos) ネットワーク認証に基づくチケット・ベースのセキュリティ・システムを利用する。カーベロスに関する更なる詳細に対しては、そのプロジェクトの刊行物を参照することができるが、その詳細は本発明の一部も形成するものではない。

【 0003 】 DCEのようなクライアント・サーバ・ネットワーク環境では、クライアント・ログイン・リクエストを検証したクライアント・アトリビュートの登録を維持するセキュリティ・サーバ又はTGS(チケット賦与サーバ) が、クライアントの特権アトリビュートの詳細を表す特権チケット賦与チケット(PTGT) をクライアントに発行する。しかる後、クライアントは、その

10

20

30

40

50

3

ネットワーク上のアプリケーション・サーバによるサービスを求めるリクエストによって、セキュリティ・サーバにこのチケットを与えることが可能になる。そのアプリケーション・サーバは、クライアント及びそのセキュリティ・アトリビュートを認証する更なるチケット又はキーをそのクライアントに戻す。そのチケット又はキーは、サービスを求めるリクエストによってDCEサーバに供給可能である。DCEでは、そのクライアントのアイデンティティ及びアトリビュートは、クライアントとセキュリティ及びアプリケーション・サーバとの間で送られる種々のチケットに含まれた特権アトリビュート証明(PAC)に含まれる。

【0004】そのようなシステムの更なる詳細は、プレントイス・ホール社(Prentice Hall, Inc)が1992年に発行した「OSF: DCEの紹介(Introduction to OSF DCE)」、[Open Software Foundation]という文献に見ることができる。そのようなシステムでは、PACは、サービス・リクエスト(一般には、リモート・プロシージャ・コール(RPC))に回答して発行されたチケットにおいてセキュリティ・サーバがクライアントのPACを配置することができるように、及びクライアントによりチケットを与えられるアプリケーション・サーバがリクエストを受けるべきかどうかをPACから確認できるように、DCE環境に関連したアイデンティティ及びセキュリティ・アトリビュート(それ自体及びグループのアトリビュート・セット)を含む。

【0005】DCEでは、クライアントのアイデンティティ及び特権がクライアントの特権アトリビュート証明(PAC)においてサーバに送られる。PACに含まれた情報に基づいて、サーバにより許可決定が行われる。これは、許可決定を行う時、クライアント及びサーバがDCEのためにフォーマットされたそれ自体及びグループ・セットの情報のみを使用する限り十分に働く。

【0006】異種の計算環境では、非DCE資源、即ち、DCE環境の外の資源に対するゲートウェイとして作用するDCEサーバが存在し得る。これらの資源へのアクセスは、DCE-PACを理解しない非DCEアクセス制御マネージャによって制御されるであろう。その代わりに、これらACLマネージャは、種々の形式のアイデンティティ及び特権情報、一般には、英数字のユーザID及びグループID、に基づいて許可決定を行うであろう。DCE特権情報をそのようなゲートウェイに与えるDCEクライアントは、それらクライアントがそれらのPACにおいてDCE特権アトリビュートを与えるだけなので非DCE資源にアクセスすることはできないであろう。一方、それら資源と関連したアクセス制御マネージャは、非DCE特権アトリビュートを予期し、理解する。そのようなサーバに到達するDCEクライアント・リクエストは、互換性のない認証及び許可アイデンティティ方法がそのクライアント及びサーバによって使用

4

されるため、外部資源からサービスを受けることができないであろう。

【0007】この問題を解決するためには、クライアントの非DCEアイデンティティ及び特権をそのDCEアイデンティティと関連付けるセキュリティ機構が必要である。これは、DCEクライアントがゲートウェイ・サーバを通して外部データ及び資源にアクセスすることを可能にする。

【0008】DCEクライアントの非DCE特権をDCEサーバに転送するための種々の方法が考えられた。それらの利点及び欠点について、簡単な検討を後述することにする。

【0009】第1の方法によれば、アプリケーション・サーバによって要求されるすべてのクライアント特権が、セキュリティ・サーバ上に保持されたレジストリからアプリケーション・サーバによって検索される。DCEクライアントは、アプリケーション・サーバに対してチケットをリクエストし、そのチケットをアプリケーション・サーバに正規の方法で与える。クライアントのPACはDCE特権アトリビュートしか含まない。クライアントの非DCE特権アトリビュートは、拡張アトリビュートとしてレジストリに記憶される。アプリケーション・サーバがクライアントの非DCEアトリビュートのうちのどれかを要求する場合、それは、レジストリに対する明示的なRPCを作ることによってそれらをそのレジストリから検索する。この利点及び欠点は次のようである。即ち、

利点:

(1) クライアント・コードに対して変更の必要がない。

(2) セキュリティ・サーバにおけるカーベロス・コードに対して変更の必要がない。

欠点:

(1) DCEにおいては、PACはクライアントからサーバに特権を転送する機構である。この技法はこの原理に反するであろう。

(2) パフォーマンス。アプリケーション・サーバは、クライアント・リクエストが満足される前にレジストリを照会しなければならない(即ち、リモート・プロシージャ・コール行う)。

【00010】第2の方法によれば、クライアントの非DCE特権は拡張PAC(XPAC)におけるそのDCE特権と結合することである。非DCE特権アトリビュートをXPACに配することが可能な種々の時点がある。

【0011】(a)自動的に、PTGTが最初にクライアントによって獲得される時。これは現在のDCE-PAC機構に類似している。現在のDCEにおいては、クライアントは、そのログイン処理の間にDCEグループ(即ち、そのDCE特権)の完全なセットを検索、

10

20

30

40

50

5

し、グループ・リストをそのログイン・コンテキストに配置する。その後クライアントがPTGTをリクエストする時(ログイン・シーケンス中のその後の時点において、或いは、それがその第1サーバ・チケットをリクエストする時)、そのグループ・リストは新しいPTGTに対するリクエストと共に提起される。それらグループは特権サーバによって検証され、PACの形でPTGTに配置される。この自動的特権検索の方法はクライアントの拡張特権にも適用可能である。しかし、拡張特権を組み込むことは、クライアントがログイン時にそのすべての拡張特権を検索すること及びそれらをPTGTリクエストにおいて提起することを必要とする。その利点及び欠点は次のようである。即ち、

利点:

(1) カーベロスTGSリクエスト処理は変更を必要としない。

欠点:

(1) クライアント・ログイン・コードが変更されなければならない。

(2) クライアントはすべての特権を、それらが要求されているかどうかに関係なく、検索する。

(3) 十分に基準化していない。

【0012】(b) 要求に基づいて、PTGTが最初に獲得される時。これは前の方法と類似している。しかし、拡張アトリビュートは自動的に検索されない。代わりに、クライアントが非DCEサーバをアクセスすることを知った時、すべての拡張アトリビュートが検索される。この方法はクライアントにおける何らかのインテリジェンスを必要とする。クライアントは、それがXPACを欲しているかどうかを決定しなければならない。その利点及び欠点は次のようである。即ち、

利点:

(1) カーベロスTGSリクエスト処理は変更を必要としない。

欠点:

(1) クライアントは、それが非DCEサーバをアクセスすることを知らなければならない。

(2) クライアントはすべての特権を、それらが要求されているかどうかに関係なく検索する。

(3) 十分に基準化していない。

【0013】(c) 要求に基づいて、特定サーバに適合されたPTGTが獲得される時。この方法では、クライアントは特定の拡張特権だけをリクエストし(それが特定の非DCE資源をアクセスすることを欲するため)、そして、XPACにおいてこれらの特権を含む新しいPTGTのためのリクエストを提起する。その利点及び欠点は次のようである。即ち、

利点:

(1) カーベロスTGSリクエスト処理は変更を必要としない。

6

(2) 十分に基準化する。

欠点:

(1) クライアントは高度にインテリジェントでなければならない。

(2) クライアントは、それが非DCEサーバをアクセスすることを知らなければならない(いつも可能とは限らない。例えば、委任が行われている場合クライアントが最終的なターゲット・サーバについて知ることが可能でないこともある)。

10 (3) クライアントは、どのようなアトリビュートをPACに配置すべきかを知らなければならない。

(4) クライアントは、それが非DCEサーバをアクセスしたい時、新しいPTGTをリクエストしなければならない。

【0014】

【発明が解決しようとする課題】本発明は第2の方法の更なる変形を使用する。本発明の目的は、計算環境においてチケット・ベースのセキュリティ・システムを実施することにある。なお、その計算環境では、資源をアクセスするためにクライアントに対して発行されるチケットに含まれた特権許可証明又は同等のデータ・エレメントは、その環境からアクセス可能な資源をアクセスするためにクライアントにとって必要なアイデンティティ及び特権データを含む必要がある場合には拡張可能であるが、通常の許可パッケージと互換性のないセキュリティ・システムを利用する。

【0015】本発明を実施し得る例示的環境として本願を通して利用されるDCEのコンテキストにおいて、これは次のような目的を達成するように設計された拡張PAC(又は、XPAC)の利用を伴う。即ち、

(1) PACに非DCE特権アトリビュートを挿入するための機構を提供する。

(2) DCEクライアント・コードに対する変更を必要としない(それによって、DCEクライアントにおける完全な透明性を維持する)。

(3) XPACを使用しないすべてのDCEサーバ(即ち、既存の(DCE1.0)技法を使用して構築されたDCEサーバ、及び拡張PACを理解せず、それを期待もしない現在及び将来のDCEサーバ)との相互運用性を維持する。

40 (4) 既存のTGS及びセキュリティ実行時コードに対する修正を最小にする。

(5) 必要なポート・プロシージャ・コールの数を最小にする。

(6) 新しい拡張アトリビュートをシステムに容易に追加することを可能にする。

(7) 分散環境を十分に大きいものに基準化する。

【0016】XPAC機構の使用はDCE環境の構成に次のような制約を課する。即ち、

50 (1) XPACを使用するサーバを含んだセルのセキュ

7

リティ・サーバは、XPAC拡張を呼び出すことができるものでなければならない。

(2) XPACを使用したいサーバは、XPACを処理できる実行時コードを含まなければならない。

【 0 0 1 7 】

【 課題を解決するための手段】本発明によれば、クライアントは、拡張特権がそのPACに加えられるべきであることさえ知らない。TGSがアプリケーション・サーバに対してチケットを発行する時、それは、それらを必要とするアプリケーション・サーバに対してだけ拡張
10 アトリビュートを加える。同様に、TGSは、アプリケーション・サーバが必要とする特定のアトリビュートだけを加える(クライアントのアトリビュートすべてを加えるのとは対照的に)。利点はクライアントにおける全体的な透明性及び良好な基準化であり、欠点はTGSリクエスト処理が変更を必要とすることである。

【 0 0 1 8 】本発明によれば、セキュリティ・サーバ及び少なくとも1つのアプリケーション・サーバを含むタイプの分散型計算環境における改良が提供される。そのセキュリティ・サーバは、クライアント・アイデンティ
20 ティ及びアトリビュートとその環境における資源に関連したクライアント特権とに関するデータを含んだアトリビュート・レジストリを有する。アプリケーション・サーバは、その環境の外部の資源へのアクセスを行い且つその環境のセキュリティ要件とは互換性のないセキュリティ要件を持った資源へのアクセスを行う。又、セキュリティ・サーバは、その環境内のサーバによるサービスを必要とするクライアントに対して、要求に応じてチケットを発行する。それらチケットはコード化データを含む
30 特権アトリビュート証明を有し、その特権アトリビュート証明は、その環境内のクライアントのアイデンティティ及び特権アトリビュートに関する情報を与えるように、サーバへの供給時にデコード可能である。本発明によるこのようなセキュリティ計算環境における改良では、セキュリティ・サーバは、前記外部の資源のうちの少なくとも1つに関するクライアント・アイデンティティ及び特権アトリビュートに関する追加情報と各外部の資源がその追加情報を必要とするというストラクチャに関するデータとを含む拡張レジストリと、サーバが外部
40 の資源へのアクセスを行うことによるサービスのためにクライアントによってリクエストされたチケットにそのような追加情報を更なるコード化データとして含むための手段とを有し、外部の資源へのアクセスを行うサーバは、更なるコード化データを認識するための手段と、そのようなデータをデコードし、その外部の資源へのアクセスのために必要なストラクチャにそれを配置するための手段とを有する。

【 0 0 1 9 】

【 発明の実施の形態】図1は本発明を組み込んだネットワークの部分におけるセキュリティ相互作用を示し、図
50

8

2はそのネットワークの関連部分の構成を示す。図2はクライアントと、ネットワークで接続された別の計算機上で走る種々のサーバとを示す。場合によっては、同じ計算機上で走るプロセスによって相異なるサーバが実行されることがあり、又その同じ計算機がDCE環境の内部及び外部の両方における資源を持つことがあり、又1つのサーバにおける多くのインスタンスが相異なる計算機上で走ることを理解すべきである。図1において、セキュリティ・サーバは、USRのようなユーザを作るために、ここではRGY-EDIT、即ち、DCEアトリビュート・レジストリ・エディタと呼ばれるプログラムを利用してクライアントのDCEアトリビュートを入れるアドミニストレータADMと相互作用する。なお、そのユーザはアプリケーション・サーバSVRにリモート・ファンクション・コールRPCを発行するアプリケーション・クライアントCLを走らせる。

【 0 0 2 0 】サーバSVRがDCEサーバである場合、それはその内容をアクセス制御リストACLと比較するためにクライアントによって与えられる特権アトリビュート証明PACを処理するであろう。ユーザUSRがログインする時、ログイン・プロセスはセキュリティ・サーバTGSにおける認証サーバにログイン・リクエストを送る。その認証サーバは、それがDCE資源へのアクセスをリクエストすることを可能にするチケットPTGTをユーザに発行する。ユーザのアプリケーション・クライアントがサーバSVRの資源にアクセスする必要がある場合、それはその目的のためのチケットをセキュリティ・サーバTGSにリクエストする。そのセキュリティ・サーバは、クライアントが供給するためのPACを含むサーバ・チケットをサーバSVRに与える(ユーザが適正な特権を有するものと仮定して)。以下の事項はすべて正規のDCEオペレーションに適用する。

【 0 0 2 1 】本発明によって与えられる拡張は、図2において概略的に示されたネットワークN1との関係において更に後述される。図2では、DCEネットワークはゲートウェイ・サーバGSを含み、そのゲートウェイ・サーバは非DCEサーバRSを、図示のような第2の非DCEネットワークN2によって或いは同じ計算機内に設けることによってアクセス可能である。各サーバは、そのネットワーク又はローカル及びネットワーク・オペレーティング・システムOSによるネットワークに接続され、各セキュリティ・サーバ及びアプリケーション・サーバは、ネットワークから受け取ったリクエストのセキュリティ事項を処理するためのセキュリティ・ルーチン・プロセスSRを有するであろう。

【 0 0 2 2 】説明される本発明の実施例の中心的な特徴は拡張PAC又はXPACである。PACは、DCEクライアントに適用するDCEアイデンティティ及び特権アトリビュートを含んだデータ・ストラクチャである。PACは、カーペロス・チケットとして一般に知られた

チケットの許可データ・フィールドにおいてクライアントからサーバに送られる。許可データは許可データ・エレメントのリストより成る。PACはこれらエレメントの1つであり、正規のDCEでは、それは許可データにおける唯一のエレメントである。

【0023】PACは簡単にはチケットにコピーされない。それは、先ず、ピッケル(pickle)され、しかる後、許可データに変換される。PACをピッケルすることはPACデータ・ストラクチャを平滑化し、ネットワークを通して転送可能なフォーマットにそれらフィールドを変換することである。これは、チケットが暗号化される前に行われる。ピッケルされた結果は、ピッケルと呼ばれることが多い。

【0024】チケットがDCEサーバに到達する時、サーバのセキュリティ実行時モジュールは許可データからPACをアセンブルし直す責任がある。暗号化されたチケットは暗号解読され、許可データはピッケルされたPACに変換され、しかる後、アンピッケル(unpickle)されなければならない。PACをアンピッケルすることは、ピッケル・プロセスのアクションを逆にすることに関連する。PACの送信されたフォーマットはデータ・ストラクチャ・フォーマットに戻し変換される。

【0025】XPACは、PACと同じ方法でクライアントからサーバに転送される。それはピッケルされ、チケットの許可データ・フィールドにおける単一の許可データ・エレメントとして送られる。許可データ・エレメントそのものは、単一のピッケル(基本DCEのピッケルされたPAC)ではなく、ピッケルのリスト(基本DCEのピッケルされたPACに続く一組の追加のピッケルであり、各追加のピッケルは拡張アトリビュートを表す)を保持するように拡張される。

【0026】特権及びアイデンティティは、各セキュリティ機構が異なって定義するエンティティである。DCEクライアントのアイデンティティは、ローカル・エリア・ネットワークのような他の計算環境におけるクライアントのアイデンティティとは異なる形式で表示される。しかし、アイデンティティ及び特権を表示する方法に関係なく、本発明は、DCEクライアントがその種々のアイデンティティ及び特権のすべてをXPACにおいて与えることを可能にする。これを達成するために次のようなオブジェクトが使用される。即ち、
特権アトリビュート・オブジェクト
特権アトリビュート・ハンドラ
特権アトリビュート・マネージャ
特権オブジェクト
これらのオブジェクトの各々については、更に詳しく後述することにする。

【0027】XPAC設計における特権の基本単位は特権アトリビュート・オブジェクトである。このオブジェ

クトは3つの情報片、即ち、アトリビュート・タイプ、アトリビュート・エンコーディング、及びアトリビュート値を含む。アトリビュート・エンコーディングはアトリビュートをピッケルに変換する方法を指定する。2つの一般的なアトリビュート・タイプ、即ち、単純アトリビュート及び複雑アトリビュートがある。単純アトリビュートは、単一のエンティティより成るアトリビュート値を含む。単純アトリビュートは、エンコーディング/デコーディング機能のデフォルト・セットを使用してエンコード及びデコード可能である。例えば、単純アトリビュートは、単一の文字ストリング(又は、単一の整数)を含むアトリビュート値を持つものでよい。アトリビュートは、文字ストリング(又は、整数)エンコーディング/デコーディングを使用してエンコード/デコードされるであろう。複雑アトリビュートは如何ように複雑であってもよい。複雑アトリビュートのアトリビュート値は文字ストリング、整数、及びバイト・ストリームの組合せを含むものでよい。複雑アトリビュートは、アトリビュート値のフォーマットの知識を持つカスタマイズされた機能によってのみエンコード/デコード可能である。これらのカスタマイズされた機能はアトリビュート・ハンドラ(後述のアトリビュート・ハンドラの説明参照)において与えられる。

【0028】上述のように、アトリビュート値に含まれた情報は、そのアトリビュートが単純アトリビュートであるか或いは複雑アトリビュートであるかによって異なるであろう。単純アトリビュートに対しては、アトリビュート値は単一の情報片であり、そのアトリビュートから直接に抽出可能である。複雑アトリビュートに対しては、その値は多くの情報片を含むことがある。アトリビュート・ハンドラは、種々の情報片を抽出する方法を知る機能を提供しなければならない。

【0029】複雑アトリビュートの例は、2つの情報片、即ち、ユーザIDを表す文字ストリングとグループIDを表す文字ストリングのリストとを含む。このアトリビュートを供給されるアトリビュート・ハンドラは、そのアトリビュートに含まれる任意の個々の情報片(例えば、ユーザID)を抽出するために使用される抽出機能を与えるであろう。

【0030】DCE環境において使用されるべき複雑特権アトリビュートは、そのアトリビュートを定義及び操作するために使用される一組の定義及び機能によっても達成されなければならない。特権アトリビュートに適用する定義及び機能のパッケージは特権アトリビュート・ハンドラと呼ばれる。

【0031】特権アトリビュート・ハンドラは次のものを具体化する。即ち、(1)そのハンドラによりサポートされるアトリビュートをサーバのアトリビュート・マネージャに登録するために使用される初期設定機能、及び(2)アトリビュート・クラスについての情報を得る

11

ために、又はアトリビュート・クラスのインスタンスを処理及び操作するために利用可能な機能を含む機能ベクトル。

【0032】初期設定機能はアトリビュート・クラスについての情報を戻す。これはアトリビュート・クラス識別子(名前及びUUI D)及びアトリビュート・クラス機能ベクトルを含む。アトリビュート・クラスのための機能ベクトルは次のことを行う機能を含む。即ち、

- (1) アトリビュート・クラスのインスタンスを作成する。
- (2) アトリビュート・クラスを識別するUUI Dを戻す。
- (3) アトリビュート・クラスを識別する文字ストリング名を戻す。
- (4) アトリビュート・インスタンスに含まれた情報を印刷する。
- (5) アトリビュート・インスタンスをピックアップする。
- (6) アトリビュート・インスタンスをアンピックアップする。
- (7) アトリビュート・インスタンスから情報を抽出する。
- (8) アトリビュート・インスタンスに置かれるべき情報に対してレジストリを照会する。
- (9) アトリビュート・インスタンスに割り振られた資源を自由にする。
- (10) アトリビュート・インスタンスから以前に取り出された情報を自由にする。

【0033】各アトリビュート・ハンドラはそのアトリビュート・クラスのインスタンスを処理するために上記機能のそれ自身のカスタマイズされたバージョンを与えるであろう。

【0034】特権アトリビュート・ハンドラPAH(図2)は、TGSによって及びターゲット・アプリケーション・サーバGSによって使用される。TGSは、特権アトリビュート・ハンドラに与えられた機能を使用してレジストリからアトリビュートの種々の部分を検索し、アトリビュートを形成し、しかる後、それをピックアップする(従って、それはカーベロス・チケットのXPAC部分に配置可能である)。ターゲット・アプリケーション・サーバはハンドラ機能を使用して、XPACから抽出後にアトリビュートをアンピックアップしそしてそのアトリビュートから特定の情報を抽出する。アトリビュート・ハンドラ機能は、アプリケーション・サーバ・コード又はTGSコードによって直接に呼び出されることはないことに留意すべきである。それらは、XPACセキュリティ実行時API(アプリケーション・プログラミング・インターフェース)及びXPACアプリケーション・サーバAPIの下で呼び出される。これは、アプリケーション・サーバ及びTGSロジックをアトリビュート・ハンドラ機能から切り離す。

12

【0035】一般に、特権アトリビュート・ハンドラは(静的に又は動的リンク・ライブラリ(DLL)を介して)アプリケーション・サーバ・コードとリンクされる。チケット許可セキュリティ・サーバの場合には、例外が生じる。TGSの場合、アトリビュート・ハンドラは、セキュリティ・サーバが初期設定する時、動的にサーバに付加される。これは、TGSがTGSコードを再コンパイル及び再リンクする必要なしにアトリビュート・ハンドラを追加又は削除することを可能にするために行われる。アトリビュート・ハンドラがセキュリティ・アトリビュートを操作するため、セキュリティアドミニストレータだけがセキュリティ・サーバにアトリビュート・ハンドラを加えることを許可されなければならない。

【0036】特権アトリビュート・マネージャPAM(図2)は、どのクラスの複雑アトリビュートがサーバに対して定義されるかを表す情報を記憶するコンポーネントである。各サーバは1つの特権アトリビュート・マネージャ・コンポーネントを含む。

【0037】サーバが複雑アトリビュートを操作できる前に、それはアトリビュート及びその関連のハンドラをアトリビュート・マネージャに登録しなければならない。(アトリビュート登録は、サーバ初期設定時に行うことが可能である)。初期設定機能のアドレスをアトリビュート・マネージャに送ることによって、アトリビュートが登録される。この機能はアトリビュート・ハンドラによって広められる。アトリビュート・マネージャは、アトリビュート・クラスに関する次のような情報、即ち、アトリビュート・クラスUUI D、アトリビュート・クラス名、及びアトリビュート・クラス機能ベクトル、を取得しそして記憶するために初期設定機能を使用する。

【0038】UUI D及び名前はアトリビュート・クラスを識別し、機能ベクトルはアトリビュート・クラスのインスタンスを操作するのに必要な機能を与える。アトリビュート・マネージャはアトリビュート・ハンドラからこの情報をすべて取得し、それを内部のルックアップ・テーブルに記憶する。

【0039】アトリビュートがサーバによって処理されるべき場合、そのサーバは、まず、そのアトリビュートが単純アトリビュートであるか又は複雑アトリビュートであるかを決定しなければならない。そのアトリビュートが単純アトリビュートである場合、デフォルト・ルーチンが使用可能である。そのアトリビュートが複雑アトリビュートである場合、サーバはアトリビュート・マネージャを照会してその複雑アトリビュートが登録されているかどうかを決定する。それが肯定される場合、アトリビュート・マネージャは、その後そのアトリビュートを処理するために使用可能な機能ベクトルをサーバに戻す。その複雑アトリビュートが登録されていない場合、

アトリビュート・マネージャはエラー・ステータスを戻し、サーバはそのアトリビュートを無視しなければならない。

【0040】特権オブジェクトは特権アトリビュートのコンテナである。XPACは、DCEアトリビュート及び非DCEアトリビュートを含む特権オブジェクトである。例えば、XPACはDCE部分及び1つのローカル・ホスト・セキュリティ・サブシステム特権アトリビュートを含むことができる。特権オブジェクトの観点から、それが含む特権アトリビュートは不透明なデータ・オブジェクトである。特権オブジェクトは任意の数の特権アトリビュート・オブジェクトを含むことができる。

【0041】図2は、XPAC拡張を使用するクライアント・サーバ交換に関連の主要なコンポーネントを示す。DCEサーバにアクセスすることを欲するDCEクライアントはDCE TGSからそのサーバに対するチケットをリクエストする。そのTGSは、ターゲット・サーバが非DCE特権機構と関連のサーバであるかどうかを決定するためにその拡張レジストリを調べる。非DCE特権機構と関連したターゲット・サーバは、クライアントがPACの代わりにXPAC(DCE特権に加えて非DCE特権も含む)を与えることを要求する。ターゲット・サーバがXPACを要求する場合、TGSは拡張レジストリからそのクライアントの適当な拡張アトリビュートを得て、XPACを形成する。

【0042】結局、クライアントはネットワークN1を介してサーバにサービス・チケットを与える。サーバは、PACに対するプロシージャと同じプロシージャを呼び出すことによってクライアントのXPACに対するポインタを得る。そこで、サーバは、後述のAPIを使用してXPACから種々の拡張アトリビュートを抽出することができる。

【0043】DCEクライアントの観点からは、変化は全くない。非DCE資源へのアクセスをリクエストするクライアントは、その資源へのアクセスを制御するDCEサーバにチケットをリクエストする。クライアントはそのPTGTをTGSに送り、それに応答してサーバ・チケットを受け取る。

【0044】クライアントが受け取ったチケットは、正規のDCE-PACではなくXPACを含む。これはクライアントにとって透明である。結局、クライアントがターゲット・サーバを呼び出す時、それはXPACを含むサーバ・チケットを送る。

【0045】アドミニストレータがクライアント及びターゲット・サーバ自体に対する拡張レジストリにおいて拡張アトリビュート情報を構成した後には、XPACは形成可能ではなく、使用可能でもない。拡張特権アトリビュートを使用するターゲット・サーバに対して、アドミニストレータは、ターゲット・サーバが使用する各拡張アトリビュートに対するサーバ特権機構レコードを

定義しなければならない。サーバ特権機構は、サーバへのアクセスに必要な特権アトリビュートのセット及び各特権アトリビュートに適用するオプション追加データを含むようにターゲット・サーバのレジストリを論理的に拡張する。

【0046】そのセットにおける各特権アトリビュートはUIDによって指定される。このUIDは、レジストリから直接に検索されそしてクライアントのXPACに置かれるべき単一の単純アトリビュートのUIDであるか、或いはアトリビュート・ハンドラが形成しそしてXPACに挿入する複雑アトリビュートのUIDであろう。XPACに配置されるべき特権アトリビュートが何らかの方法で修飾されなければならない場合、この目的のためには追加のデータを使用することが可能である。

【0047】特権機構及び追加のデータの使用を、次のような例によって最も良く説明することができる。

【0048】サーバ1は、クライアントがそのサーバの非DCE資源へのアクセスを得る前に、そのクライアントがそのアクセス・コードをXPACにおけるそのローカル・ホストに与えることを必要とする。サーバ1は計算機A、B、C、D、及びEに存在する。計算機A、B、及びCにおけるサーバ1のインスタンスはタイプA1のアトリビュートを使用し、一方、計算機D及びEにおけるサーバ1のインスタンスはタイプB1のアトリビュートを使用する。タイプA1及びB1のアトリビュートは両方とも単純アトリビュートである。従って、必要なデータ・ストラクチャ、即ち、単純なデフォルト・ストラクチャが知られるように、XPACは単純アトリビュートとして識別されるそのクライアントのタイプA1及びB1アトリビュートを含まなければならない。

【0049】サーバ2は、外部資源へのゲートウェイ・アクセスを与えるサーバである。これらの資源にアクセスするために、クライアントは、ユーザ・プロファイル全体を含む(ユーザID、グループ・リスト、及び他のセキュリティ・データを含む)複雑アトリビュートを与えなければならない。単純アトリビュートのリストとして個々のアトリビュートすべてを指定する代わりに、複雑な特権アトリビュートA2が定義される。アトリビュートA2のインスタンスはその値フィールドにユーザ・プロファイルを含む。A2は、A2のアトリビュート・ハンドラがセキュリティ・サーバ及びターゲット・サーバの両方において導入される場合のみ使用可能である。A2のハンドラは、XPACに及びXPACからユーザ・プロファイルを封入し及び抽出する方法を知っているコードである。アドミニストレータは、サーバ1及びサーバ2に対するレジストリにおける次表に示すデータを指定するであろう。即ち、

【表1】

サーバ自体	15 必要な特権機構	追加データ
サーバ1	A 1 B 1	A、B、C D、E
サーバ2	A 2	なし

【 0 0 5 0 】追加データとラベル付けされたフィールドは、XPACに配置されるべき情報を更に修飾するために使用可能なオプション・エントリである。このフィールドにおける情報のフォーマット及び意味は、それが適用するアトリビュートのコンテキストにおいてのみ理解可能である。上記の表1において、追加データ「A、B、C」はアトリビュートA1のコンテキストにおいてはローカル計算機A、B、及びCを意味するように解される。

【 0 0 5 1 】RGY-EDITを使用するアドミニストレータは必要な特権機構を文字ストリングとして入れる。これらはUIDに変換される。追加データは、それが適用するアトリビュート・タイプにとって意味のあるフォーマットで入れられる。例えば、追加データが計算機のリストである場合、それも文字ストリングのリストとして入れることが可能である。これらの文字ストリングは内部的にUIDに変換可能である。又、アドミニストレータは、上記ターゲット・サーバをアクセスする各クライアントに対して必要な拡張特権アトリビュートをすべて入れなければならないであろう。クライアントの主要なエントリは次のように思われる。即ち、クライアント自体:

ベースDCEアトリビュート

拡張アトリビュート --->A1-A-値1

A1-B-値2

A1-C-値3

B1-D-値4

B1-E-値5

【 0 0 5 2 】拡張特権アトリビュートをセキュリティ・レジストリ・データベースDBに加えるための機構が必要である。適当な機構の一例は、オープン・ソフトウェア・ファウンデーションから入手可能なDCE RFC 6.0 において提案された「拡張レジストリ・アトリビュート(ERA)」である。以下では、この必要な機構をERAと呼ぶことにする。そのERA機構は、DCEアドミニストレータによって呼び出され、拡張サーバ及びクライアント・アトリビュートERAをサーバ及びクライアント・レジストリ・エントリDB(図1)に加える。このERA機構は、外部APIを使用して、及びDCE管理ユーティリティ(例えば、RGY-EDIT)を介してアクセス可能でなければならない。同様に、TGSは、レジストリからの拡張特権アトリビュートを検索するためにERAに対するアクセスを持つであろう。

16

【 0 0 5 3 】無修正のDCEでは、TGSがサーバ・チケットに対するクライアント・リクエストを受け取る時、それは受信したPTGT(カーボス許可データ・フィールドにPACを含む)を暗号解読及びデコードし、PTGTから宛先サーバ・チケットに許可データを盲目的にコピーし、しかる後、その結果のチケットを再エンコード及び再暗号化する。

【 0 0 5 4 】XPAC設計は2つのローケーション、即ち、セキュリティ・サーバ初期設定及びTGSリクエスト処理、におけるセキュリティ・サーバに変更を強いる。セキュリティ・サーバが先ずスタートする時、それは余分の構成ファイルを読み取らなければならない。このファイルはファイル名のリストを含む。各ファイル名は複雑な特権アトリビュートに対するアトリビュート・ハンドラを含むファイルを表す。セキュリティ・サーバは、各アトリビュート・ハンドラに含まれた機能を動的にロードする。アトリビュート・ハンドラPAHを動的にロードすることによって、新しいアトリビュートが、再コンパイル又は再リンクの必要なしにそのシステムに追加可能である。ハンドラを含むファイル名を構成ファイルに追加すること、TGSを停止させること、しかる後、TGSを再開始させることによって、(新しい、更新されたハンドラのリストがロードされ)、新しいアトリビュートが追加可能である。これの代替え方法の1つは、構成ファイルの管理がRGY-EDITの枠組に統合される場合、TGSを停止させることなく構成ファイルを更新しそして新しいハンドラをロードするのである。

【 0 0 5 5 】TGSがアトリビュート・ハンドラをロードする時、それは特権アトリビュート・マネージャPAMIに、そのハンドラを登録する。その後、TGSがハンドラの機能をアクセスする必要がある時、アトリビュート・マネージャはそのアクセスを行うであろう。

【 0 0 5 6 】TGSに対する第2の変更では、PACを保持した入力許可データがXPACを保持した出力許可データに変えられるように、TGSリクエスト処理がわずかに修正される。サーバ・チケットに対するクライアント・リクエストがTGSに到達する時、現在のDCEにおけるように、TGSリクエスト処理は正規のデコーディング及び暗号解読オペレーションから始まる。しかし、PTGTからサーバ・チケットに許可データをコピーする前に、TGSは2ステップのルックアップ・プロシージャを開始する。その第1ステップでは、TGSは、ターゲット・サーバが何れかの非DCEサーバ特権機構と関連しているかどうかを決定するためにERA機構(上記参照)を使用する。そのようなサーバ特権機構が存在しない場合、TGSは、正規のTGSリクエスト処理を使用してターゲット・サーバ・チケットを発行する。しかし、何れかの非DCEサーバ特権機構がターゲット・サーバに適用する場合、TGSはそのルックアップの第2ステップを開始する。

17

【0057】ルックアップの第2ステップは、各非DCEサーバ特権機構に対して適用するクライアントの拡張アトリビュートを検索することに関連する。各特権機構は、ターゲット・サーバに与えられなければならない拡張特権アトリビュートのクラスを識別する（即ち、特権機構UI Dはアトリビュート・クラスUI Dに等価である）。各機構に対して、TGSは、まず、そのクラスのアトリビュートに対するハンドラが存在するかどうかを決定するために特権アトリビュート・マネージャを照会する。それが存在する場合、特権アトリビュート・マネージャは、そのアトリビュート・クラスに適用する機能ベクトルを戻す。TGSはこのベクトルを使用して、そのクライアントに属し且つそのサーバにおいて適用するアトリビュートを検索する。その機能ベクトルは、再びそのアトリビュートをピックアップするために使用され、TGSはそのピックアップを許可データ・フィールド（基本DCE特権を含んでいる）に付加し、そしてサーバ・チケットはそのクライアントに戻される。

【0058】アトリビュート・ハンドラが存在しない場合、TGSは、必要なアトリビュートが単純アトリビュートであると仮定し、クライアントのレジストリ・エントリからそのアトリビュートを検索するためにERA機構を使用する。そのアトリビュートはデフォルト・ピックアップ・ルーチンを使用してピックアップされ、TGSはサーバ・チケットにおける許可データ・フィールドにそのピックアップを付加し、そしてサーバ・チケットはクライアントに戻される。

【0059】拡張特権を持った許可データ・フィールドのフォーマットは次のように見える。即ち、許可データ・カテゴリ（OSF-DCE）

長さ（基本及び特別ピックアップを含む）

内容――> 基本DCEピックアップ

アトリビュート1ピックアップ

アトリビュート2ピックアップ

...

アトリビュートNピックアップ

【0060】拡張特権アトリビュートを必要とするアプリケーション・サーバは、それがXPACを扱わなければならないことを知っている。そのサーバは、それが必要とする拡張アトリビュートのタイプも知っている。XPACに含まれた情報をサーバが使用するためには、そのサーバは、それが使用するアトリビュートのタイプを登録しなければならない、そしてXPACに含まれた拡張アトリビュートから所望の情報を取り出さなければならない。

【0061】サーバが拡張アトリビュート・タイプを登録する時、それは、セキュリティ実行時モジュールがそのアトリビュート・タイプを処理するに必要な情報を与える。この情報はそのアトリビュートに対する初期設定機能のアドレスである。サーバがアトリビュートを登録

18

する時、そのサーバに対する特権アトリビュート・マネージャは初期設定機能を使用してそのアトリビュートに対するクラス情報を取得し、そして記憶する。この情報はアトリビュート・クラス指示子（UI D）、アトリビュート・クラス・ストリング名、及びそのクラスからアトリビュートのインスタンスを操作するために使用されるルーチンを含む機能ベクトルを含む。ターゲット・サーバの特権アトリビュート・マネージャは、TGSが記憶するのと同じ方法でこの情報を内部ルックアップ・テーブルに記憶する（図1参照）。

【0062】XPACを含むチケットがサーバに到達する時、そのサーバの実行時セキュリティ・モジュールはカーベロス許可データを分析し、XPAC特権オブジェクトを形成する。その分析中、それは、まず、DCE特権を抽出する。拡張（即ち、DCEピックアップに付加された追加のピックアップ）が存在する場合、それは各拡張のクラスを調べ、そのクラスのための機能ベクトルに対する特権アトリビュート・マネージャを照会する。特権アトリビュート・マネージャがそのクラスを登録している場合、その機能ベクトルは戻され、ベクトルのアンピックアップ・ルーチンが呼び出され、そしてアンピックアップされたアトリビュートが特権オブジェクトに加えられる。アトリビュート・マネージャがそのクラスに対するハンドラを登録していない場合、アトリビュートは無視される。アプリケーション・サーバは、前述の外部APIの1つを呼び出すこと及び所望のアトリビュートを抽出することによって、拡張特権をアクセスすることができる。DCE1.0サーバは、XPACに含まれた拡張特権を無視するであろう。それらはXPACをDCE1.0PACのように扱うであろう。

【0063】XPACに対応する特権オブジェクトは次のような形式を持つであろう。即ち、

基本DCE部分

拡張UI D

アトリビュートの数

アトリビュート1――>A1フィールド

アトリビュート2――>A2フィールド

【0064】拡張UI Dは、セキュリティ実行時モジュールにとって良く知られたUI Dである。それは、基本DCE部分に続く拡張アトリビュートの存在を表す。UI Dの不存在は、古いスタイルのPACが処理されようとしていることを表し、従って、セキュリティ実行時モジュールは、何れの拡張も処理しようとはしないであろう。

【0065】次の事項は拡張PAC処理に関連した主要ステップである。

（1）TGSが始動し、すべての特別の特権アトリビュート・ハンドラをロードする。

（2）サーバSが始動し、それが認識するすべての特別のアトリビュートを登録する。

(3) クライアント C が TGS にサーバ S のチケットをリクエストする。

(4) TGS は、サーバ S が何らかの拡張特権アトリビュートを必要としているかどうかをチェックする。

(5) それが肯定される場合、TGS はそれらアトリビュートのインスタンスに対してクライアント C のレジストリ・エントリを照会し、それらをサーバ・チケットに挿入する。

(6) TGS がサーバ S に対するチケットをクライアント C に戻す。

(7) クライアント C がサーバ S にリクエストを送り、サーバ S にチケットを渡す。

(8) サーバ S のセキュリティ実行時モジュールがそのチケットから XPAC を抽出する。

(9) サーバ S は XPAC から種々のアトリビュートを明瞭にリクエストし、それらを必要に応じて使用する。

【 0 0 6 6 】 正規の DCE では、他のセルにおけるサーバをアクセスすることを望んでいる 1 つのセルにおけるクライアントは、それら 2 つのセルにおける DCE セキュリティ・アドミニストレータによって信頼関係が形成されている場合、そのように行うことができる。サーバのセルにおける TGS がそのクライアントに対するサービス・チケットを、起点クライアントの PAC を使用して発行することができるので、これは可能である。

【 0 0 6 7 】 しかし、1 つのセルにおけるクライアントが他のセルにおけるサーバをアクセスすることを望み且つそのサーバがそのクライアントの拡張アトリビュートを必要とする場合、そのモデルは更に複雑になる。そのサーバがそのクライアントの拡張アトリビュートを必要とすることを知っている唯一のエンティティはそのサーバのセルにおける TGS である。この TGS は、そのクライアントに属する拡張アトリビュートを検索できなければならない。クライアント・セル・セキュリティ・サーバが拡張特権アトリビュートをサポートすることを仮定することはできない。事実、そのサーバのセルは、外部のクライアントをそのセルに相互登録するための機構を与えなければならない、そしてこれら外部のクライアント・エントリに拡張アトリビュートを加えなければならない。

【 0 0 6 8 】 この設計は、ERA 機構が外部クライアント自体及びそれらのアトリビュートを 1 つのセルに相互登録するための機構を与えることを仮定している。このような機構の場合、XPAC を必要とする外部セルにおいて生じたリクエストは次のような方法で処理されるであろう。即ち、

(1) サーバのセルにおける TGS が始動し、すべての特別な特権アトリビュート・ハンドラをロードする。

(2) サーバ S が始動し、それが認識するすべての特別なアトリビュートを登録する。

(3) そのサーバのセルにおけるアドミニストレータが

そのセルにクライアント C を相互登録し、サーバ S に適用するそのクライアントの拡張アトリビュートを加える。

(4) クライアント C がそれ自身の TGS にサーバ S のチケットをリクエストする。

(5) マルチ・セル相互作用が開始され、その結果、サーバ S のチケットに対するリクエストがそのサーバの TGS に行われる。

(6) この TGS は、サーバ S が何らかの拡張特権アトリビュートを必要としているかどうかをチェックする。

(7) それが肯定される場合、TGS はそれらアトリビュートのインスタンスに対して外部クライアント自体 C のための相互登録されたエントリを照会し、それらをサーバ・チケットに挿入する。

(8) TGS がサーバ S に対するチケットをクライアント C に戻す。

(9) クライアント C がサーバ S にリクエストを送り、サーバ S にチケットを渡す。

(1 0) サーバ S のセキュリティ実行時モジュールがそのチケットから XPAC を抽出する。

(1 1) サーバ S は XPAC から種々のアトリビュートを明瞭にリクエストし、それらを必要に応じて使用する。

【 0 0 6 9 】 この実施例は、XPAC を認識しないすべてのサーバとの相互運用性を維持する。これは可能なことである。それは、サーバがアドミニストレータによって適正に登録されると仮定すると、XPAC を認識するサーバだけがそれらを受け取り、XPAC を認識しないサーバはサーバ・チケットにおいてそれらを受け取らないためである。たとえ、アドミニストレータがサーバを、XPAC を認識するものとして不正確に登録しても、サーバは XPAC における拡張を無視するであろうし、XPAC を DCE 1.0 PAC として扱うであろう。

【 0 0 7 0 】 この実施例は DCE セキュリティ・サービスにおいて包含するための新しい API を提案する。その設計は既存の基本 DCE コードに対する非常にわずかな変更を必要とするだけであり、基本 DCE に対する次のような変更でもって基本 DCE に組込可能である。即ち、

(1) 特権アトリビュート・ハンドラをロードするためのセキュリティ・サーバ始動コードにおけるフック。

(2) アトリビュートをチケットに加えるための TGS 処理コードにおけるフック。

(3) 入力 XPAC をアンパックするためのセキュリティ実行時モジュールにおけるフック。

【 0 0 7 1 】 アクセスされるべき特定の資源にとって特有の特権アトリビュート処理コードのすべては、動的にロードされる (TGS によって) か、或いは静的にリンクされる (アプリケーション・サーバによって) 。それ

は、アトリビュート・ハンドラ・ルーチンに対して指定されたフォーマットに適合する必要がある。拡張アトリビュートを利用することを望んでいるアプリケーション・サーバは新しいAPI を呼び出す。

【 0 0 7 2 】 本願において開示されるXPAC設計は、PACに加えられるべきDCEクライアントの非DCE特権アトリビュートのための機構を与える。その設計の特性は次のように要約することができる。即ち、

(1) クライアント側のコードに対するコード変更は必要なく、その機構はクライアントにとって透明である。

(2) クライアントは、ターゲット・サーバの性質に関する明瞭な知識を維持する必要がない。

(3) XPACを認識しないサーバは影響されない。

(4) セキュリティ・サーバのTGSに対して、及びXPACを使用することを欲するアプリケーション・サーバのセキュリティ実行時モジュールに対して、わずかな修正が必要なだけである。

(5) 追加のリモート・プロシージャ・コールは導入されない。

(6) 新しいアトリビュート・ハンドラをロードすることによって、追加の拡張アトリビュートが適応可能である。

(7) XPACに配置された拡張アトリビュートは特定のターゲット・サーバへのアクセスのために必要なものだけである。

【 0 0 7 3 】 A. セキュリティ・サーバのためのAPI

(a) 特権アトリビュート・ハンドラをロードする
パラメータ：なし

概説：このAPI は、それが処理できる特権アトリビュートのリストをロードするためにTGSによって呼び出される。

高レベル・フロー：

・ <アトリビュート・ハンドラ・モジュールのファイル名>を構成ファイルから読み取る。

・ アトリビュート・ハンドラの初期設定ルーチン(登録機能)をそのモジュールから動的にロードする。

・ 特権アトリビュート・ハンドラを登録するためのルーチン(下記参照)を呼出し、初期設定ルーチンのアドレスを入力として渡す。

【 0 0 7 4 】 (b) 許可データを付加する

*40

パラメータ：

入力

許可データ 一組の付加ピッケルより成るカーベ
ロス許可データ

出力

PAC XPACのアンピッケル・バージョン

概説：この内部機能はアプリケーション・サーバのセキュリティ実行時モジュールによって呼び出される。それは1つ又は複数のピッケルされた特権項目を含むカーベロス許可データを変換し、その特権をアンピッケル

パラメータ：

入力

クライアント TGSリクエストを行うクライアントの名前

サーバ TGSリクエストのターゲットであるサーバの名前

入力-出力

許可データ クライアントの許可データ新しい特権が付加される

10 *概説：これは、TGSが入力のTGSリクエストをデコードし且つ暗号解読した後、TGSによって呼び出される内部機能である。そのリクエストにおけるターゲット・サーバが、クライアントが特別な非DCE特権を与えることを必要とするものである場合、この機能はこれらの特権を検索し、許可データに含まれた既存の特権にそれらを付加する。

高レベル・フロー：

・ 入力の許可データを調べ、DCE許可データである第1許可データ・エレメントを決定する。

20 ・ クライアント及びサーバ名をUIDに変換する。

・ ターゲット・アプリケーション・サーバが非DCE特権機構を使用するかどうかを決定する。

・ サーバによって要求される各特権機構に対して、
-アトリビュートが登録されたハンドラ機能を有するかどうかを決定する

-アトリビュートがハンドラを有する場合、

-アトリビュートのインスタンスを作成するためにその作成機能呼び出す

-クライアントの拡張レジストリ・エントリからアトリビュート値を検索するためにその照会レジストリ機能呼び出す

-アトリビュートが登録されたハンドラを持たない場合、

-実際のアトリビュートに関してレジストリを照会する

-アトリビュートをピッケルする

-許可データに保持されたピッケルのセットにそのピッケルを付加する

【 0 0 7 5 】 B. セキュリティ実行時機能

(a) 拡張PACを形成する

し、XPACを形成する。この機能は既存の許可データ・ツリー・PAC処理を置換する。

高レベル・フロー：

50 ・ カーベロス許可データを一組のピッケルされたアトリ

ビュートに変換する

- ・ DCE 部分をアンピックルする
- ・ 各追加のピックルに対して、
- ーアトリビュートをアンピックルする
- ーアトリビュートをXPAC特権オブジェクトに挿入する
- ・ 完成したXPACを戻す

【0076】実行時機能は、入力XPACに対してスペースを割り振るため及びこのスペースが最早必要ない時にはこのスペースを自由にするためにも、特権アトリビュート・マネージャに対するルックアップ・テーブルを初期設定しそしてアクセスするためにも、そのようなテーブルが最早必要ない時にはそのテーブルと関連した資源を開放するためにも、そして特権アトリビュート・マネージャを登録及び登録解除するためにも与えられる。

【0077】C. アプリケーション・サーバに対するAPI

(a) 特権アトリビュート・ハンドラを登録する

パラメータ:

入力

登録情報 特権アトリビュートに対する初期設定機能のアドレスを含む不透明なデータに対するポインタ

出力

アトリビュート・クラス 登録されたアトリビュートを識別するUUID

概説: このAPI は、それが認識しそして処理する特権アトリビュートを登録するために、サーバによって呼び出される。

高レベル・フロー:

- ・ 登録情報から初期設定機能を抽出する
- ・ アトリビュートを登録するために特権アトリビュート・マネージャを呼びだし、その機能ベクトル、クラスUUID、及びクラス名を戻す
- ・ クラスUUIDを起呼者に戻す

【0078】(b) 特権アトリビュート・ハンドラを登録解除する

パラメータ:

入力

アトリビュート・クラス 登録解除されるべきアトリビュート・クラスを識別するUUID

概説: このAPI は、それが前に登録した特権アトリビュートを登録解除するためにサーバによって呼び出される。

高レベル・フロー:

- ・ アトリビュートを登録解除するために特権アトリビュート・マネージャを呼び出す

【0079】(c) カーソルを初期設定する

パラメータ:

出力

カーソル 初期設定されたカーソルに対するポインタ

概説: このAPI は特権アトリビュート・オペレーションにおけるその後の使用のためにカーソルを初期設定する

高レベル・フロー:

カーソル・オブジェクトを割り振る

カーソルを初期設定する

【0080】(d) カーソルをリセットする

パラメータ:

入力・出力

カーソル カーソル・オブジェクトに対するポインタ

概説: このAPI は既存のカーソルをリセットする。これは特権アトリビュートの照会を再開させるために行われる。

高レベル・フロー:

- ・ カーソル・オブジェクトをリセットする

【0081】(e) カーソルを削除する

パラメータ:

出力

カーソル カーソル・オブジェクトに対するポインタ

概説: このAPI は既存のカーソルによって保持された資源を自由にする

高レベル・フロー:

- ・ カーソル・オブジェクトを削除する

【0082】(f) XPACから基本DCE-PACを抽出する

パラメータ:

入力

特権 クライアントのXPACに対するポインタ

出力

DCE特権 XPACから取り出された基本DCE-PACに対するポインタ

概説: このAPI はXPACから基本DCE-PACを抽出する。出力はXPACに含まれたDCE-PACのコピーである

高レベル・フロー:

- ・ DCE-PACのコピーに対してメモリを割り振る
- ・ DCE-PACを新たに割り振られたメモリにコピーする

【0083】(g) 基本DCE-PACと関連の資源を自由にする

25.

パラメータ:

入力

特権 自由にされるべき資源を有する基本DCE
-PAC

概説: このAPI は、先行のAPI に対する呼出しを介して得られた基本DCE -PACと関連した資源を自由にする

高レベル・フロー:

- ・ DCE -PACを自由にする

【0084】(h) 拡張PACと関連の資源を自由にする 10

パラメータ:

入力

特権 自由にされるべき資源を有する拡張PAC

概説: このAPI は拡張PACと関連した資源を自由にする。

高レベル・フロー:

- ・ XPACの動的に割り振られた部分を自由にする
- ・ XPACの残りを自由にする

【0085】(i) 特権アトリビュートを抽出する 20

パラメータ:

入力

特権 クライアントのXPACを対するポインタ
アトリビュート・タイプ XPACから取り出さ
れるべき特権アトリビュートのタイプ

入力・出力

カーソル カーソル・オブジェクトに対するポ
インタ

出力

アトリビュート 取り出された特権アトリビュ
ートに対するポインタ 30

概説: このAPI はXPACから特権アトリビュートを抽出する。アトリビュート・タイプ入力パラメータが指定される場合、XPACはその指定されたタイプに適合したアトリビュートを見つけるために走査される。適合した特権アトリビュートに関するサーチはカーソル位置から開始する。アトリビュート・タイプ入力パラメータが「NULL」にセットされる場合、次の特権アトリビュート(カーソルの後)は戻される。抽出が成功すると、カーソル位置はその取り出された特権アトリビュートを参照するために更新される。

高レベル・フロー:

- ・ 送られたPACがXPACであることをチェックする。

- ・ カーソル位置で開始する

- ・ カーソルが過去の最後のアトリビュートを指す場合、エラーを戻す

- ・ 次のようにループする

- 一次のアトリビュートを得る

- 一次アトリビュート・タイプが指定される場合、アトリビ 50

26

ュート・タイプに関して一致するかどうかをチェックする

- カーソル位置を更新する

- 一致が見つかるまで或いはリストの終わりまでループする

【0086】(j) 特権アトリビュートのインスタンスを作成する

パラメータ:

入力

アトリビュート・タイプ 作成されるべき特権ア
トリビュートのタイプ

出力

アトリビュート 新たに作成されたアトリビュ
ートに対するポインタ

概説: このAPI は特定のクラスの特権アトリビュートのインスタンスを作成する。そのクラスの一般的なインスタンスが作成され、インスタンス特定値がその後の機能呼出しによって満たされたまま残される。この機能は、クライアントの証書に置かれるべきXPACを形成する時、セキュリティ・サーバによって呼び出される。又、それは、それが入力クライアント証書を使用してXPACを形成する時、アプリケーション・サーバのセキュリティ実行時モジュールによっても呼び出される。

高レベル・フロー:

- ・ アトリビュート・クラスが登録されているかどうかをチェックする

- ・ アトリビュート・インスタンスに対してメモリを割り振る

- ・ 後で満たされるべきすべてのインスタンス・フィールド(特定フィールド)を残して非インスタンス特定データでもってインスタンスを初期設定する

【0087】(k) アトリビュートと関連の資源を自由にする

パラメータ:

入力

アトリビュート 自由にされるべき資源を有する
アトリビュート

概説: このAPI は特権アトリビュートと関連した資源を自由にする。

高レベル・フロー:

- ・ アトリビュート・クラスが登録されているかどうかをチェックする

- ・ アトリビュートのアトリビュートークラスー特定部分を自由にする

- ・ アトリビュートと関連した資源の残りを自由にする

【0088】(l) 特権アトリビュートのタイプを抽出する

27

パラメータ:

入力

アトリビュート 特権アトリビュートに対するポ
インタ

出力

アトリビュート・タイプ 特権アトリビュートの
タイプ

概説: このAPI は特定の特権アトリビュートのタイプ
を戻す。「アトリビュート・タイプ」出力パラメータが
起呼者によって割り振られる。

高レベル・フロー:

・ 特権アトリビュートのタイプ(即ち、クラス) を表す
UI Dを戻す

【0089】(m) 特権アトリビュート から値を抽出す
る

パラメータ:

入力

アトリビュート 特権アトリビュートに対するポ
インタ

基準 複雑アトリビュートに対して、これはその
アトリビュートのどの部分が戻されるべき
かを指定する

入力・出力

カーソル カーソル・オブジェクトに対するポ
インタ

出力

アトリビュート値 特権アトリビュートに含まれ
た値

概説: このAPI は特権アトリビュート に含まれた値を
戻す。そのアトリビュート が単純アトリビュート である
場合、基準及びカーソル入力は無視される。そのアトリ
ビュート が複雑アトリビュート である場合、基準はどの
値が戻されるべきかを指定する。アトリビュート が所望
の基準の複数のインスタンスを含む場合、カーソルは次
のインスタンスを戻すために使用される。戻されたアトリ
ビュート 値によって使用されるメモリ はこの機能によ
って割り振られ、後続のAPI に対する呼出しでもって
自由にされなければならない。

高レベル・フロー:

・ 単純アトリビュート である場合、その値を戻す
・ 複雑アトリビュート である場合、それが処理可能な
ものであるかどうかを知るためにルックアップ・テー
ブルをチェックする

・ それが処理可能なものである場合、その抽出機能を
呼び出す

【0090】(n) アトリビュート 値によって使用され
る資源を自由にする

28

パラメータ:

入力

アトリビュート 特権アトリビュートに対するポ
インタ

入力・出力

アトリビュート値 自由にされるべきアトリビ
ュート値

概説: このAPI はアトリビュート 値によって使用され
る資源を自由にする

10 高レベル・フロー:

・ 単純アトリビュート の値である場合、それを自由に
する

・ 複雑アトリビュート の値である場合、それが処理可
能なものであるかどうかを知るためにルックアップ・テー
ブルをチェックする

・ それが処理可能なものである場合、その自由機能を
呼び出す

【0091】D. アトリビュート・ハンドラAPI

以下の機能はアトリビュート・ハンドラによって与えら
れる。アトリビュート・ハンドラは機能ポインタのベク
トルであり、従って、それらの機能の実際の名前は重要
ではない。そのベクトルが作成される時、それは、この
項で説明することを実施する機能に対するポインタを与
える。

(a) 作成

パラメータ:

出力

アトリビュート アトリビュート・ハンドラによ
って定義されたタイプの特権アトリビ
ュートの新たに作成されたインスタンスに
対するポインタ

30

概説: そのハンドラのタイプの特権アトリビュート のイ
ンスタンスを作成する

高レベル・フロー:

・ アトリビュートの新しいインスタンスを割り振る

・ 新しいインスタンスを指すように「アトリビュート」
出力パラメータをセットする

【0092】(b) タイプ

パラメータ:

入力

アトリビュート 特権アトリビュートに対するポ
インタ

出力

アトリビュート・タイプ アトリビュートのタイ
プを指定するUI D

概説: 起呼者によって割り振られた「アトリビュート・
タイプ」出力パラメータを、アトリビュートのタイプを
指定するUI Dに等しくセットする。「アトリビュ
ート」入力パラメータは「NULL」であってもよく、こ
れはアトリビュート・クラスのタイプが(特定のアトリ

50

ビュートに含まれるタイプとは対照的に) 出力として望ましいことを表す。

高レベル・フロー:

・アトリビュートが正しいクラスのもの(又は、NULL)であることをチェックする

・アトリビュート・クラスに対するアトリビュート・タイプに「アトリビュート・タイプ」出力パラメータをセットする

【0093】(c) 名前

パラメータ:

入力

アトリビュート 特権アトリビュートに対するポインタ

出力

アトリビュート名 アトリビュートの名前の文字ストリング表示

概説: アトリビュート・クラスの名前を「アトリビュート名」パラメータにコピーする。「アトリビュート」入力パラメータは「NULL」であってもよい。これは、アトリビュート・クラスの名前が(特定のアトリビュートの名前とは対照的に) 出力として望ましいことを表す。その名前がコピーされるバッファは起呼者によって割り振られる。

高レベル・フロー:

・アトリビュートが正しいクラスのもの(又は、NULL)であることをチェックする

・「アトリビュート名」出力アーギュメントとして送られたバッファにアトリビュート・クラスのストリング名をコピーする。

【0094】(d) プリント・アトリビュート

パラメータ:

入力

アトリビュート 特権アトリビュートに対するポインタ

概説: アトリビュートに含まれた情報をプリントする

高レベル・フロー:

・アトリビュートが正しいクラスのものであることをチェックする

・アトリビュートに含まれた情報をプリントする

【0095】(e) エンコード

パラメータ:

入力

アトリビュート 特権アトリビュートに対するポインタ

出力

エンコード・アトリビュート アトリビュートのエンコード(即ち、ピックルされた)バージョンに対するポインタ

概説: 供給されたアトリビュートをピックルする

高レベル・フロー:

・アトリビュートが正しいクラスのものであることをチェックする

・ピックルを保持するために必要なメモリを計算し、割り振る。

・アトリビュートをピックルする

【0096】(f) デコード

パラメータ:

入力

アトリビュート 情報のデコード(即ち、アンピックルされた)バージョンを満たされる特権アトリビュートに対するポインタ
エンコード・アトリビュート 特権アトリビュートのエンコード・バージョンに対するポインタ

概説:

供給されたピックルをアンピックルする

高レベル・フロー:

・アトリビュートが正しいクラスのものであることをチェックする

・ピックルをアンピックルする

・アンピックルされた情報でもってアトリビュートを満たす

【0097】(g) 抽出

パラメータ:

入力

アトリビュート 特権アトリビュートに対するポインタ

基準 抽出すべきものを指定するハンドラ特有の基準

入力・出力

カーソル アトリビュートにおける情報を横断するために使用されるカーソルに対するポインタ

出力

アトリビュート値 アトリビュートから抽出された所望の情報に対するポインタ

概説: アトリビュートから特定情報を抽出する

注意: その抽出された情報は抽出機能によって割り振られ、そしてそれはアトリビュート値を自由にする機能に対する呼出しでもって自由にされなければならない。

高レベル・フロー:

・「カーソル」がNULLである場合、アトリビュートに含まれた情報の始めにおいてサーチが開始する。

・それ以外の場合は、開始ポイントとしてカーソルを使用する

・入力基準に基づいてアトリビュートをサーチする

・基準が満たされる場合、「アトリビュート値」を保持するためのメモリを割り振り、情報を「アトリビュート

31

値」出力にコピーする

【 0 0 9 8 】 (h) 照会レジストリ
パラメータ:

入力

アトリビュート 特権アトリビュートに対するポ
インタ

概説: アトリビュートに配置されるべき情報に関してレ
ジストリを照会し、アトリビュートを満たす
高レベル・フロー:

- ・アトリビュートが正しいクラスのものであることをチ 10
ェックする
- ・アトリビュート値を得るためにレジストリに対する必
要な呼出しを行う
- ・レジストリによって戻された情報を保持するためのメ
モリを割り振る
- ・戻された情報でもってアトリビュート値を満たす

【 0 0 9 9 】 (i) アトリビュートを自由にする
パラメータ:

入力

アトリビュート 特権アトリビュートに対するポ 20
インタ

概説: アトリビュートのインスタンスに割り振られた資
源を自由にする

高レベル・フロー:

- ・アトリビュートに含まれたデータと関連した資源を自
由にする
 - ・アトリビュートそのものと関連した資源を自由にする
- 【 0 1 0 0 】 (j) アトリビュート値を自由にする
パラメータ:

入力

アトリビュート値 アトリビュートから抽出され 30
た情報に対するポインタ

概説: アトリビュートのインスタンスから抽出された情
報に割り振られた資源を自由にする

高レベル・フロー:

- ・アトリビュート値と関連した資源を自由にする

【 0 1 0 1 】 上記の機能は、それぞれ、正常終了又は1
つ或いは複数個のエラー状態を表すために値を戻すであ
らう。

【 0 1 0 2 】 上記の説明はDCEへの本発明の適用に対
して特別の関連を持っているけれども、それが、公認の
サーバ・アクセスに対して同様の許可プロシージャを有
する他の分散型計算環境においても適用可能であること
は明らかであろう。

【 0 1 0 3 】 まとめとして、本発明の構成に関して以下
の事項を開示する。

(1) クライアント・アイデンティティ及びアトリビ
ュートと環境における資源に関連したクライアント特権と
に関するデータを含んだアトリビュート・レジストリを
有するセキュリティ・サーバと、環境の外部の資源への 50

32

アクセスを行い且つ環境のセキュリティ要件とは互換性
のないセキュリティ要件を有する該環境内の少なくとも
1つのアプリケーション・サーバを含むタイプの分散
型計算環境にして、前記セキュリティ・サーバは前記環
境内のサーバによるサービスを必要とするクライアント
に対して要求に応じてチケットを発行し、前記チケット
は、前記環境内のクライアントのアイデンティティ及び
特権アトリビュートに関する情報を与えるように、サー
バへの供給時にデコード可能であるコード化データを含
む特権アトリビュート証明を有する分散型計算環境にお
いて、前記セキュリティ・サーバは、前記外部の資源の
うちの少なくとも1つに関するクライアント・アイデン
ティティ及び特権アトリビュートに関する追加情報と各
外部の資源が前記追加情報を必要とするというストラク
チャに関するデータとを含む拡張レジストリと、サーバ
が外部の資源へのアクセスを行うことによるサービスの
ためにクライアントによってリクエストされたチケット
に前記追加情報を更なるコード化データとして含むため
の手段とを有すること、及び外部の資源へのアクセスを
行うサーバは、更なるコード化データを認識するための
手段と、該認識されたデータをデコードし、外部資源へ
のアクセスのために必要なストラクチャに該デコードさ
れたデータを配置するための手段とを有することを特徴
とする分散型計算環境。

(2) 前記セキュリティ・サーバ及び前記外部の資源へ
のアクセスを行うサーバは前記セキュリティ・サーバの
前記証明における前記追加情報を含むアトリビュート・
ハンドラを含み、外部の資源へのアクセスのための構造
化データを必要とするサーバにおいて前記追加情報をデ
コード及び構造化することを特徴とする上記(1) に記
載の分散型計算環境。

(3) 前記更なるコード化データは前記環境内のクライ
アントの特権アトリビュートに関する結果のコード化デ
ータに続く単一のデータ・エレメントに含まれることを
特徴とする上記(1) に記載の分散型計算環境。

(4) サーバへのアクセスを望んでいるクライアントに
チケットを発行するためのセキュリティ・サーバと、環
境の外部の少なくとも1つの資源をアクセスすることが
できる少なくとも1つのアプリケーション・サーバとを
含むタイプの分散型計算環境に対する拡張にして、前記
チケットはクライアントのアイデンティティ及び特権ア
トリビュートに関するコード化情報を含む特権アトリビ
ュート証明を含む分散型計算環境に対する拡張機構にお
いて、前記セキュリティ・サーバにおけるデータベース
からの、クライアントのアイデンティティ及び特権ア
トリビュートに関する追加のコード化データ及びそのよう
なデータが前記環境の外部のサーバに与えられるストラ
クチャに関する追加のコード化データをストラクチャ内
に含むべく特権アトリビュート証明が拡張されるチケッ
トを発行するように前記セキュリティ・サーバを再構成

するための手段と、前記拡張された特権アトリビュート証明を認識するように、前記証明から前記追加データをデコードするように、及び前記外部のサーバに表示するためのデータを構造化するように、前記アプリケーション・サーバのセキュリティ・モジュールを再構成するた

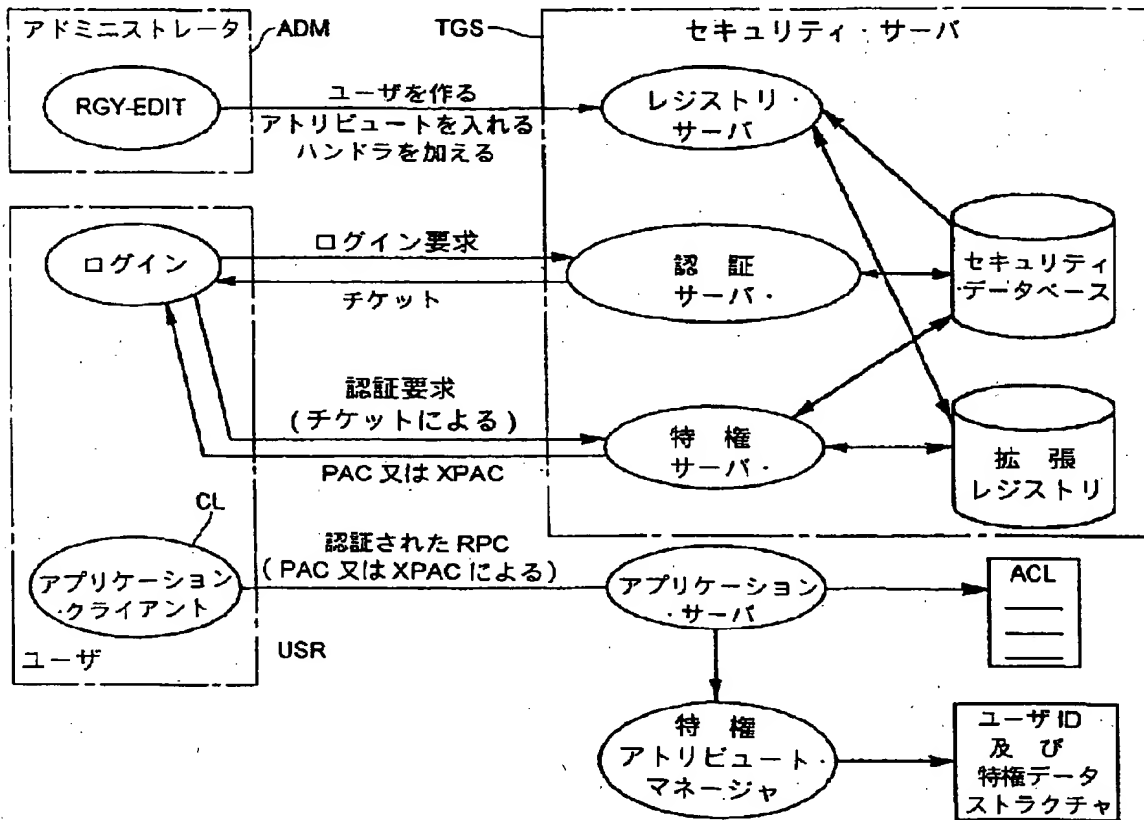
めの手段と、を含む分散型計算環境に対する拡張機構。

【図面の簡単な説明】

【図1】 アドミニストレータ、セキュリティ・サーバ、及びユーザの間の相互作用を概略的に示す。

【図2】 ネットワークの相対的部分の概略表示である。

【図1】



【 図2 】

